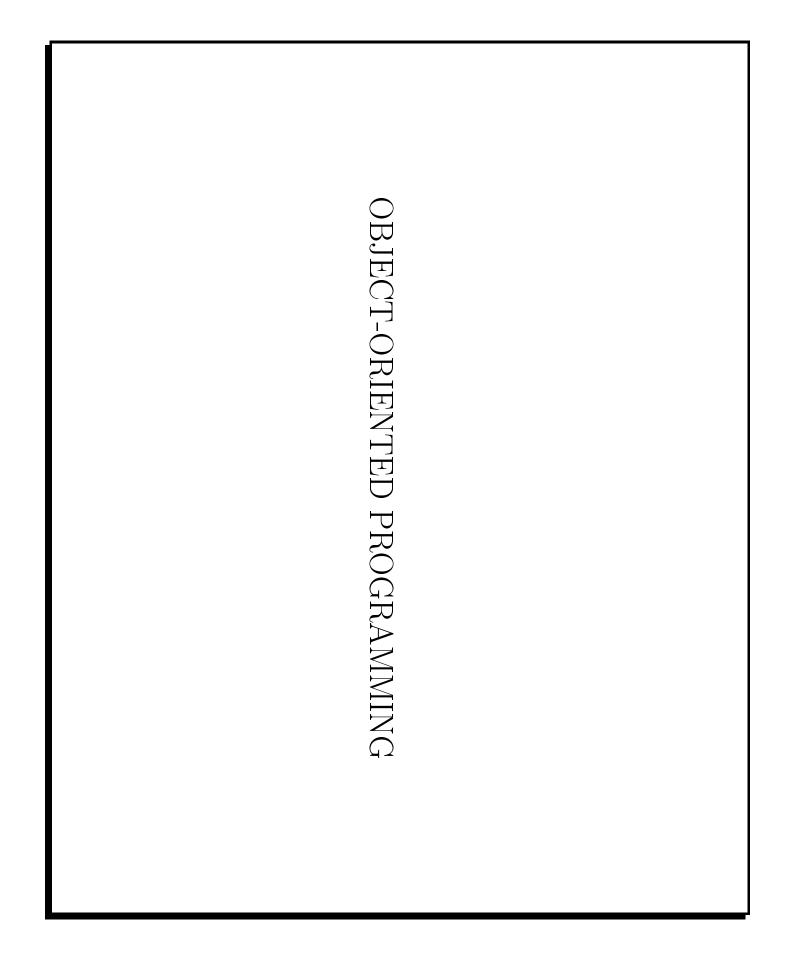
# CONSTRAINED OPTIMIZATION SOFTWARE AN OBJECT-ORIENTED FRAMEWORK FOR

Juan Meza Sandia National Laboratory

Doug Moore Rice University

May 22, 1996



# Abstraction

Abstraction is the collection of data, procedures, or both

data abstraction names a collection of data

procedural abstraction names a series of actions

object abstraction names data and the procedures that act on it

#### Examples:

```
ColumnVector v(3);
 Grid3d g(4,4,5);
                      DiagonalMatrix m(3);
                                           x = v.length();
= g.numnodes();
                        m.determinant();
```

# Encapsulation

access is restricted to the implementor of an object class. Encapulation prevents unsafe access to object data and procedures, since

```
ColumnVector v(3);
```

v.size = 
$$-2$$
;

revision, since undocumented features are inaccessible. Encapsulation prevents unauthorized access that could stifle program

Grid3d 
$$g(4,4,5)$$
;

g.storage[i\*g.size[1]] = 
$$-2$$
;

# Polymorphism

only in type. Polymorphism allows common expression of similar concepts that differ

Vector< float > v1;

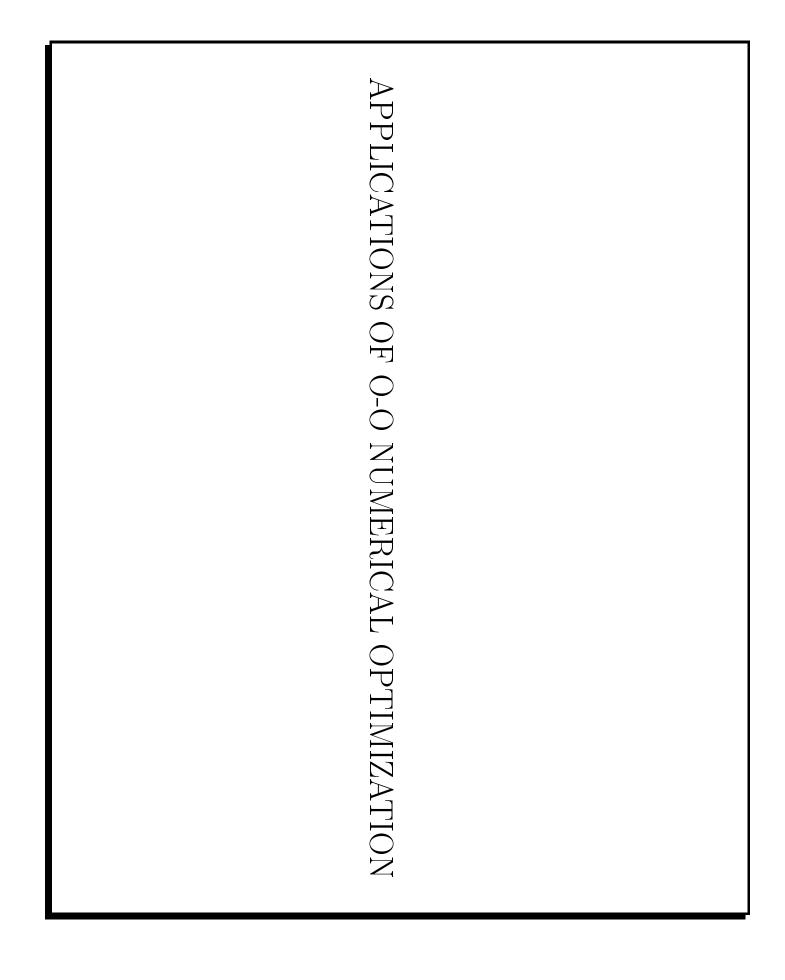
```
x = norm2(v1) + norm2(v2) + norm2(v3);
                                                                                                                                                                                                                                                                     Vector< complex< double > > v3;
                                                                                                                    Number norm2(Vector< Number > v) {
                                                                                                                                                    template<class Number>
                                                                                                                                                                                                                                                                                                   Vector< double > v2;
                                                          Number s = 0;
for (int i = 0; i < v.length(); ++i)</pre>
return sqrt(s);
                               s = s + v[i]*v[i];
```

## <u>Inheritance</u>

simpler ones. Inheritance allows complex objects to be described as extensions of

class Matrix

```
class SymmSqMatrix: public SquareMatrix
                                                                                                                                                                                             class SquareMatrix: public Matrix
                                                                                                                              float determinant();
                                                                                                                                                                                                                                                              int rank();
Vector< float > eigenvalues();
```



#### COOOL

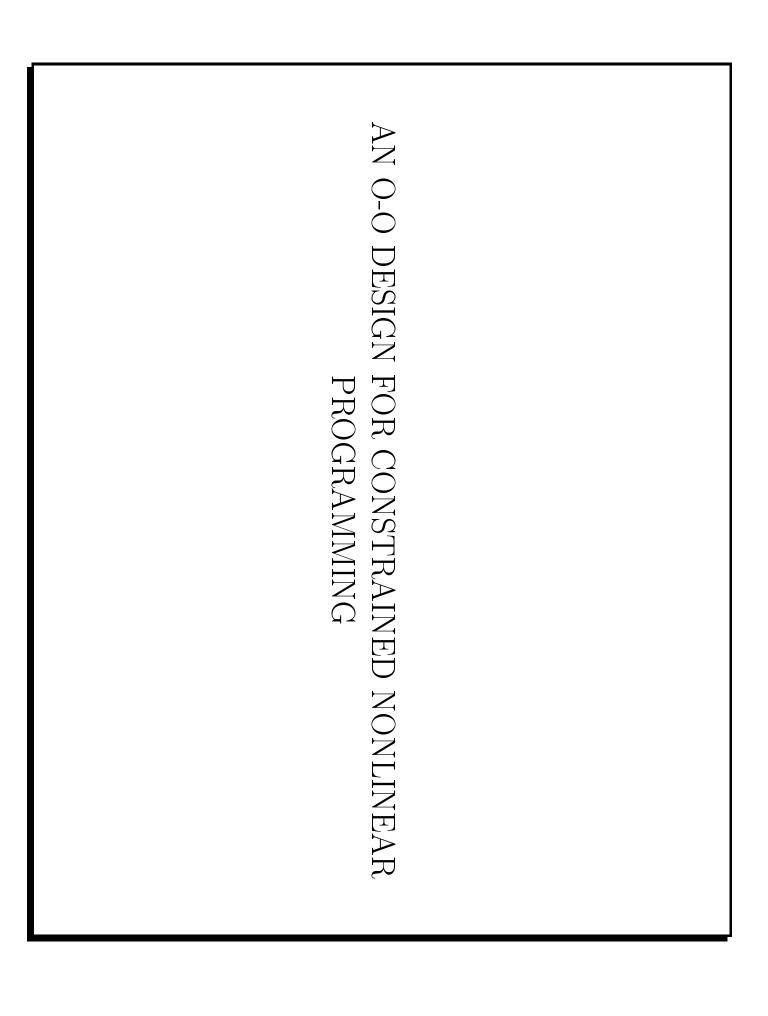
The CWP (Center for Wave Phenomena) Object-Oriented Optimization Library, by Deng, et al,

- ullet includes linear algebra and text manipulation functions
- ullet uses I/O to exchange data between optimizer and simulation
- does not use analytic derivatives
- handles only unconstrained problems
- has solved seismic inversion problems

#### OPT++

The OPT++ Nonlinear Optimization Library, by Meza,

- $\bullet$  uses the newmat08 C++ matrix library for linear algebra
- receives function data by traditional function calls
- can use analytic derivatives, but does not require them
- handles bound and inequality constraints
- has solved problems in molecular conformation, chemical vapor deposition furnace design, and structural dynamics.



#### Overview

**Problems** are defined by an OPT++ customer, and supply information about the objective and constraint functions.

Models are local approximations to problems.

Methods systematically search for feasible, optimal problem solutions.

**Tolerances** determine when a tentative solution should be accepted.

#### Problems

requires that the user specify at least The problem class is made for users to specialize, by inheritance. It

- the number of problem variables
- the number of problem constraints
- the objective and constraint function values for a given argument

The user may choose to specify

- analytic objective gradients for a given argument, in lieu of a finite difference approximation
- objective Hessians for a given argument,
- constraint Jacobians for a given argument, or
- a combination of the above in a single function call

#### Models

problem models Model and its derived classes provide several kinds of local, low-degree

Models provide a current argument and corresponding objective and constraint values

**Linear Models**, derived from Models, also provide objective gradients and constraint Jacobians as supplied by the problem.

Secant Models, like Linear Models, but approximate using secant method.

Quadratic Models, derived from Linear Models, also provide objective Hessians from the problem

QuasiModels, like QuadraticModels, but approximate inverse Hessian using update methods

#### Methods

A Method queries and updates models.

OptMethod requires its derived classes to provide an optimize method.

NewtonMethod Implements optimize by a Newton method, independent of how the Model is calculated.

CauchyMethod Implements optimize with a steepest descent method

**DirectMethod** Implements optimize with a direct search method.

### Tolerances

method, decides when to conclude computation A tolerance, given information about the progress of the optimization

maxIterations An upper limit on the number of times a Model can be updated.

fcnDelta A lower limit on the function value improvement between consecutive Model updates.

argDelta A lower limit on the distance between consecutive trial solutions.

## Availability

To learn more about the version 1.5 of OPT++, which supports bound http://midway.ca.sandia.gov/~meza/. and equality constraints, see the OPT++ web page at

15